An introduction to valued constraint satisfaction

Thomas Schiex

An introduction to Valued Constraint Satisfaction

Thomas Schiex
INRA, Toulouse, France

with contributed slides by P. Jeavons (Univ. Oxford), M. Cooper (Univ. Toulouse) Javier Larrosa (UPC, Spain), S. de Givry, D. Allouche & A. Favier (INRA, France), R. Dechter (UCI, USA), R. Marinescu (4C, Ireland)

Valued Constraint Satisfaction

- What is it and why do we need it?
- Can it be done efficiently?
- Search
- Problem transformations
- Open problems

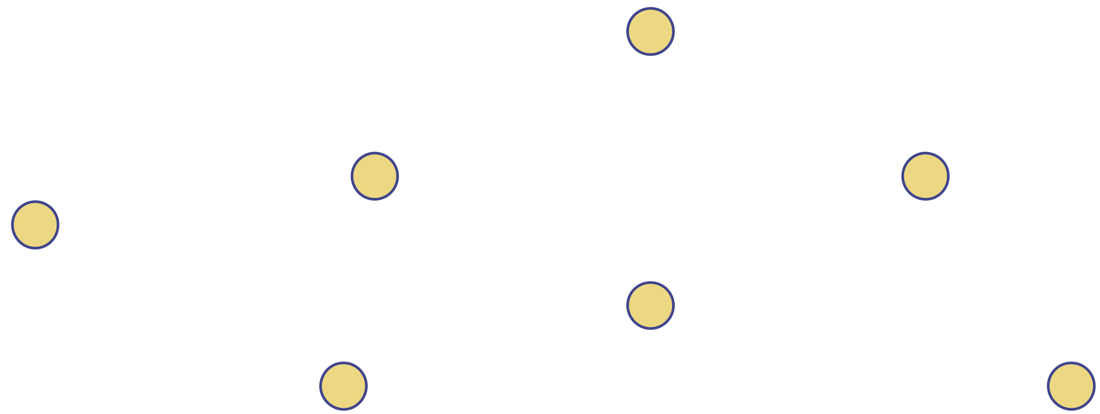
Chapter 1. What is it?

Motivation,

Definitions,

Some general theorems

Constraint Satisfaction Problem
A unifying abstraction



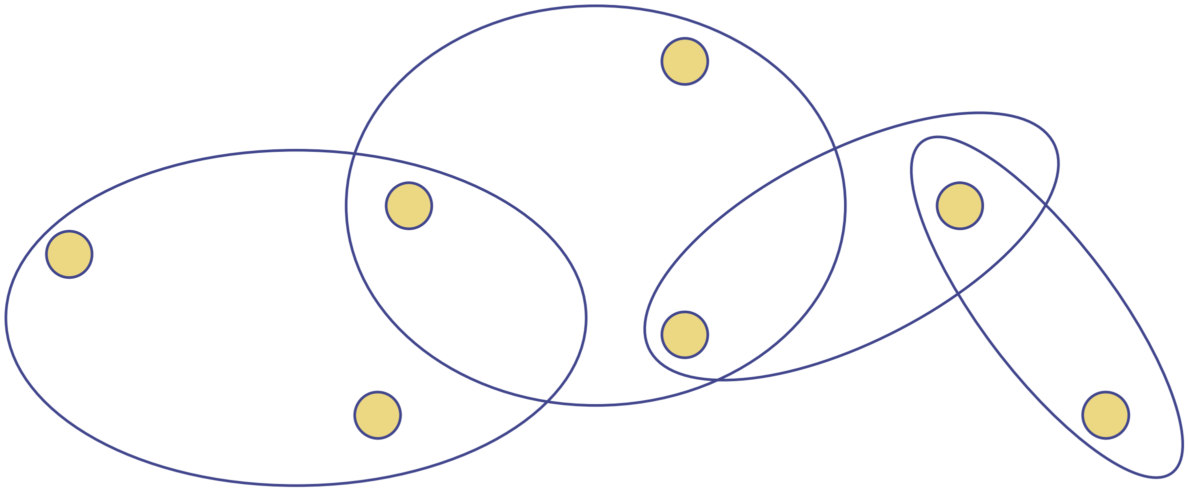
Variables = Talks to be scheduled at conference

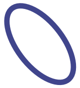
Transmitters to be assigned frequencies

Amino acids to be located in space

Circuit components to be placed on a chip

A unifying abstraction



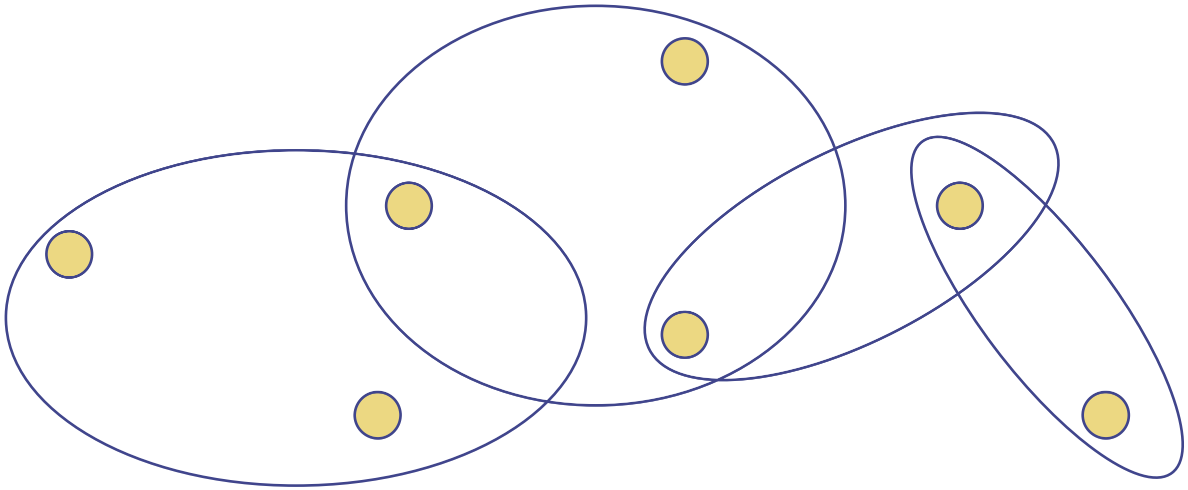
Constraints  = All invited talks on different days

No interference between near transmitters

$x + y + z > 0$

Foundations dug before walls built

A unifying abstraction



A solution is an assignment of values to variables that satisfies all the constraints

Constraint programming (OR, Ilog Solver...)

But what if…

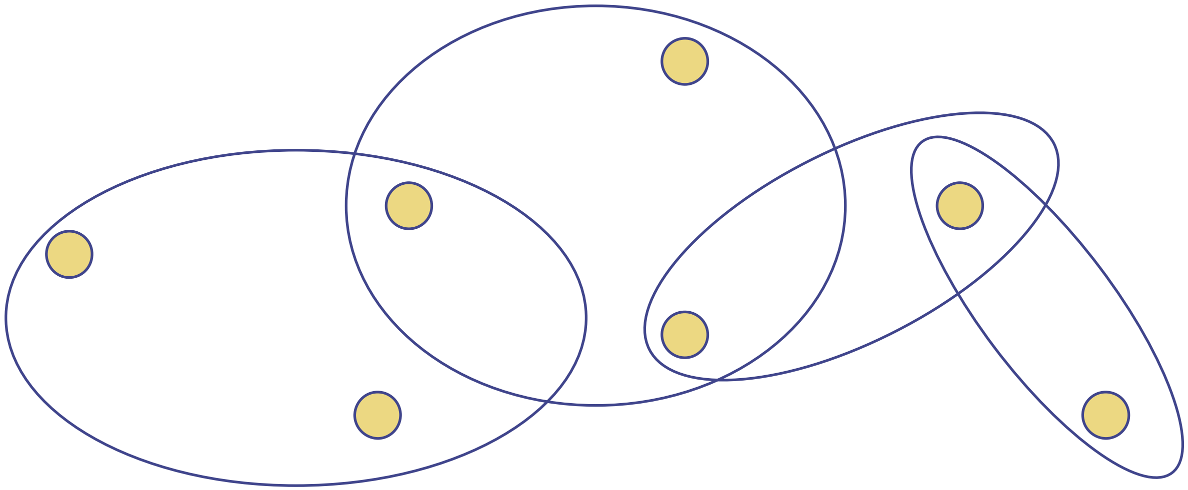
- There are lots of solutions, but some are better than others?
- There are no solutions, but some assignments satisfy more constraints than others?
- We don’t know the exact constraints, only probabilities, or fuzzy membership functions?
- We’re willing to violate some constraints if we can get a better overall solution that way?

Fragmentation/Heterogeneity

- Fuzzy CSP (easier to solve, Rosenfeld 76)
- Max, Weighted, Partial CSP (Shapiro 81, Freuder 91)
- Weighted Max-SAT
- Constraint Optimization Problems
- Lexicographic CSP
- Hierarchical Constraint Logic Programming (Borning et al)

- Pseudo-Boolean Optimisation
- Bayesian Networks
- Random Markov Fields
- Factor Graphs
- Integer Programming
- 2D grammars...

A unifying abstraction



“Constraints” associate costs with each assignment

A solution is an assignment of values to variables that minimises the combined costs

Definition of a VCSP instance

(IJCAI 1995)

- a set of n variables X_i with domains d_i
- a set of e cost functions, each having a
  - scope (list of variables)
- cost functions map assignments to costs

It only remains to specify what the possible costs are, and how to combine them

Definition of a valuation structure

- a set S of costs
- a total order $<$
- minimum and maximum elements:

  we denote these by 0 and ∞
- an aggregation operator $\oplus$ which is commutative, associative, monotonic, and such that $\forall \alpha,\ \alpha \oplus 0 = \alpha$

Examples of valuation structures

- If $S = \{0, \infty\}$, then VCSP ≡ CSP
- If $S = \{0, 1, 2, \ldots, \infty\}$, and $\oplus$ is addition,
  then VCSP generalizes MAX-CSP
- If $S = [0,1]$, and $\oplus$ is max, then VCSP ≡ Fuzzy CSP
  - If $S = \{0, 1, \ldots., k\}$, and $\oplus$ is bounded addition $+_k$
    where $\alpha +_k \beta = \min\{k, \alpha+\beta\}$,
    then VCSP ≡ Weighted CSP

Families of valuation structures

A valuation structure is idempotent if
$\forall\alpha,\ \alpha\oplus\alpha=\alpha$

All idempotent valuation structures are equivalent to Fuzzy CSP

(as in CSP redundancy of information is fine)

Families of valuation structures

A valuation structure is strictly monotonic if
  $\forall\alpha<\beta, \forall\gamma<\infty, \ \alpha\oplus\gamma < \beta\oplus\gamma$

A valuation structure is fair if
  aggregation has a partial inverse, that is,
    $\forall\alpha\geq\beta, \exists\gamma$ such that $\beta\oplus\gamma=\alpha$

All strictly monotonic valuation structures can be embedded in a fair valuation structure

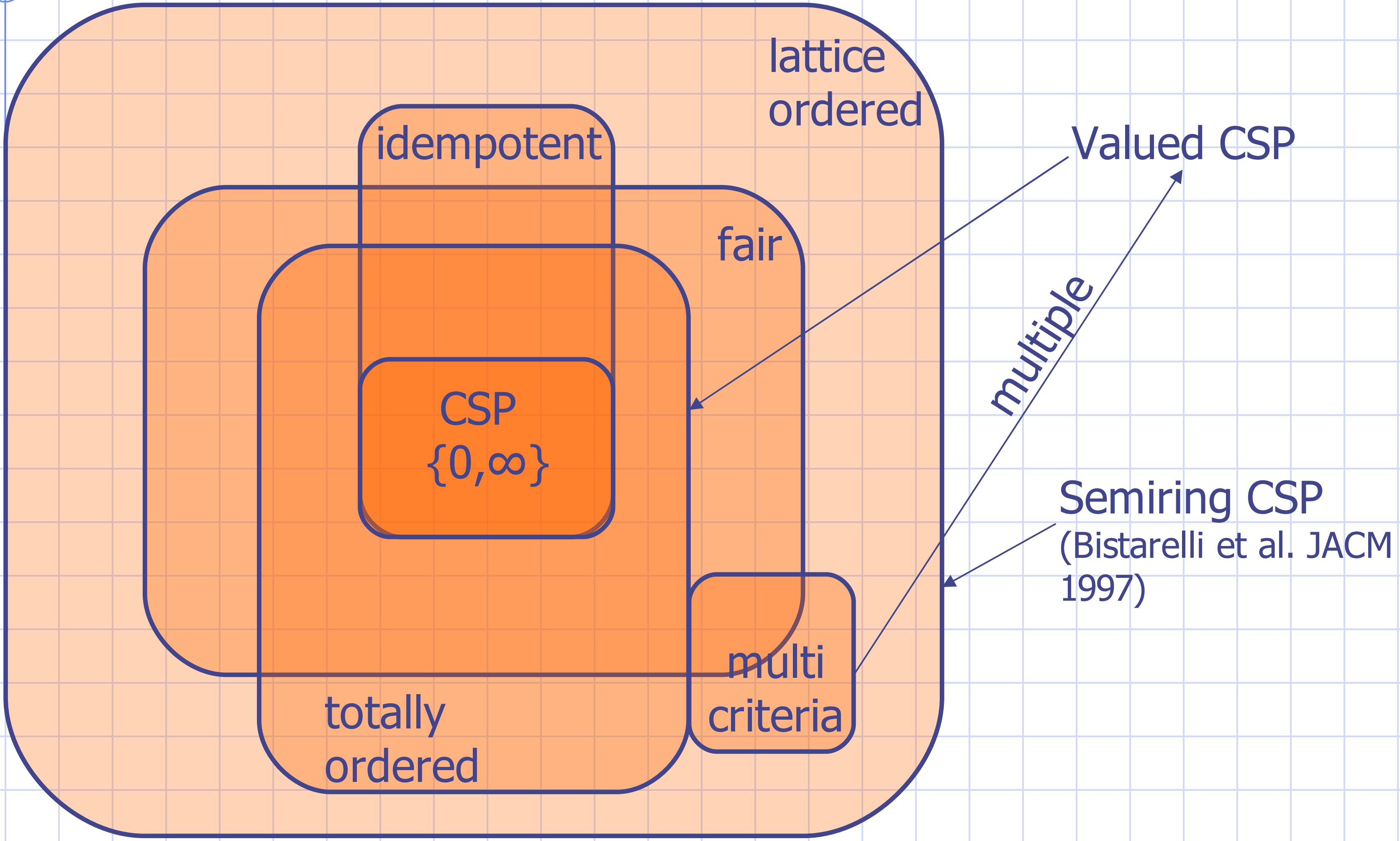
Families of valuation structures

A valuation structure is discrete if between any pair of finite costs there are finitely many other costs

All discrete and fair valuation structures
can be decomposed into
a contiguous sequence of valuation structures
with aggregation operator $+_k$

(interacting as fuzzy CSP)

General frameworks and cost structures



Bibliography

- For general background on VCSP and other formalisms for soft constraints, see the chapter on "Soft Constraints" by Meseguer, Rossi and Schiex, in the *Handbook of Constraint Programming*, Elsevier, 2006.
- For classification results on valuation structures see "Arc Consistency for Soft Constraints", *Cooper & Schiex,* AIJ, 2004.

Chapter 2. Efficiency

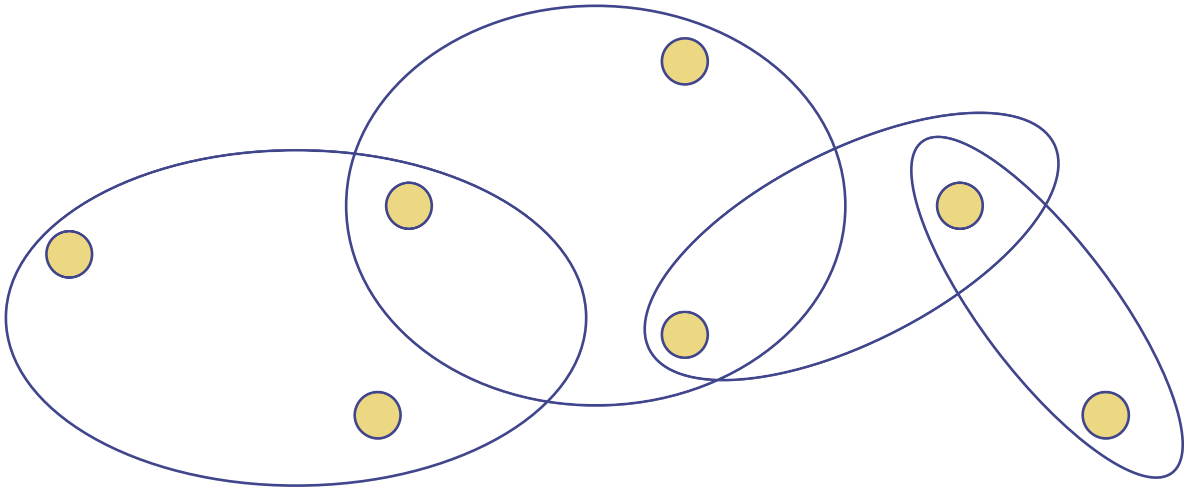
Structural restrictions,
Valued constraint languages

General question

Having a unified formulation allows us to ask *general* questions about efficiency:

When is the VCSP tractable?

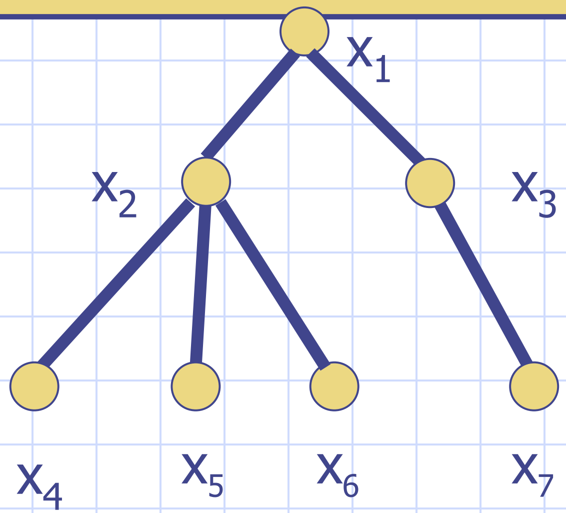
Problem features



- This picture illustrates the constraint *scopes*
- The set of scopes is sometimes called the *constraint hypergraph*, or the *scheme*
- Restricting the scheme can lead to tractability, as in the standard CSP

Structural tractability

Tree-structured binary VCSPs are tractable



Time complexity $O(e\ d^2)$
Space complexity $O(n\ d)$

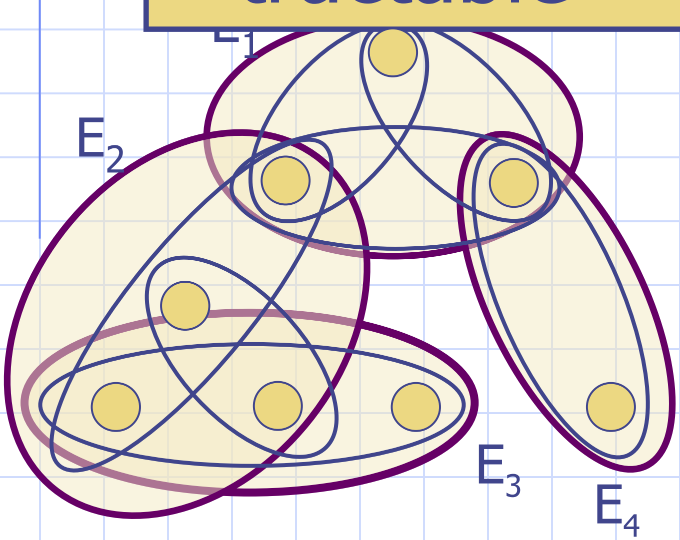
n: number of variables
d: maximum domain size
e: number of cost functions

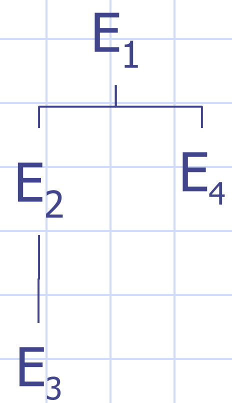
Proceed from the leaf nodes to a chosen root node

Project out leaf nodes by minimising over possible assignments

Tree decomposition

Bounded treewidth VCSPs are tractable





Time complexity $O(e\ d^{w+1})$
Space complexity $O(n\ d^{s})$

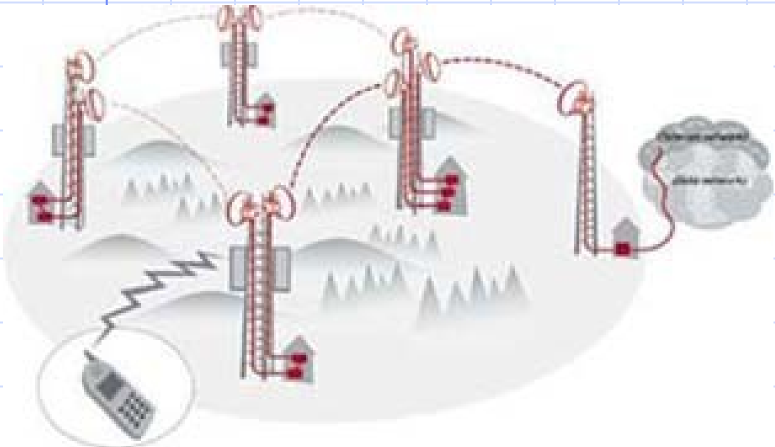
w: bounded treewidth
$= \max |Ei| - 1$

s: $\max \{|E_i \cap E_j| : i \neq j\}$

Finding a tree decomposition with minimum w* is NP-hard!

Radio Link Frequency Assignment Problem

(Cabon et al., *Constraints* 1999) (Koster *et al.*, *4OR* 2003)



- Given a telecommunication network
- …find the best frequency for each communication link, avoiding interferences

- Best can be:
  - Minimize the maximum frequency, no interference (max operator)
  - **Minimize the global interference (sum operator)**
- Generalizes graph coloring problems: $|f_i - f_j| \geq a$

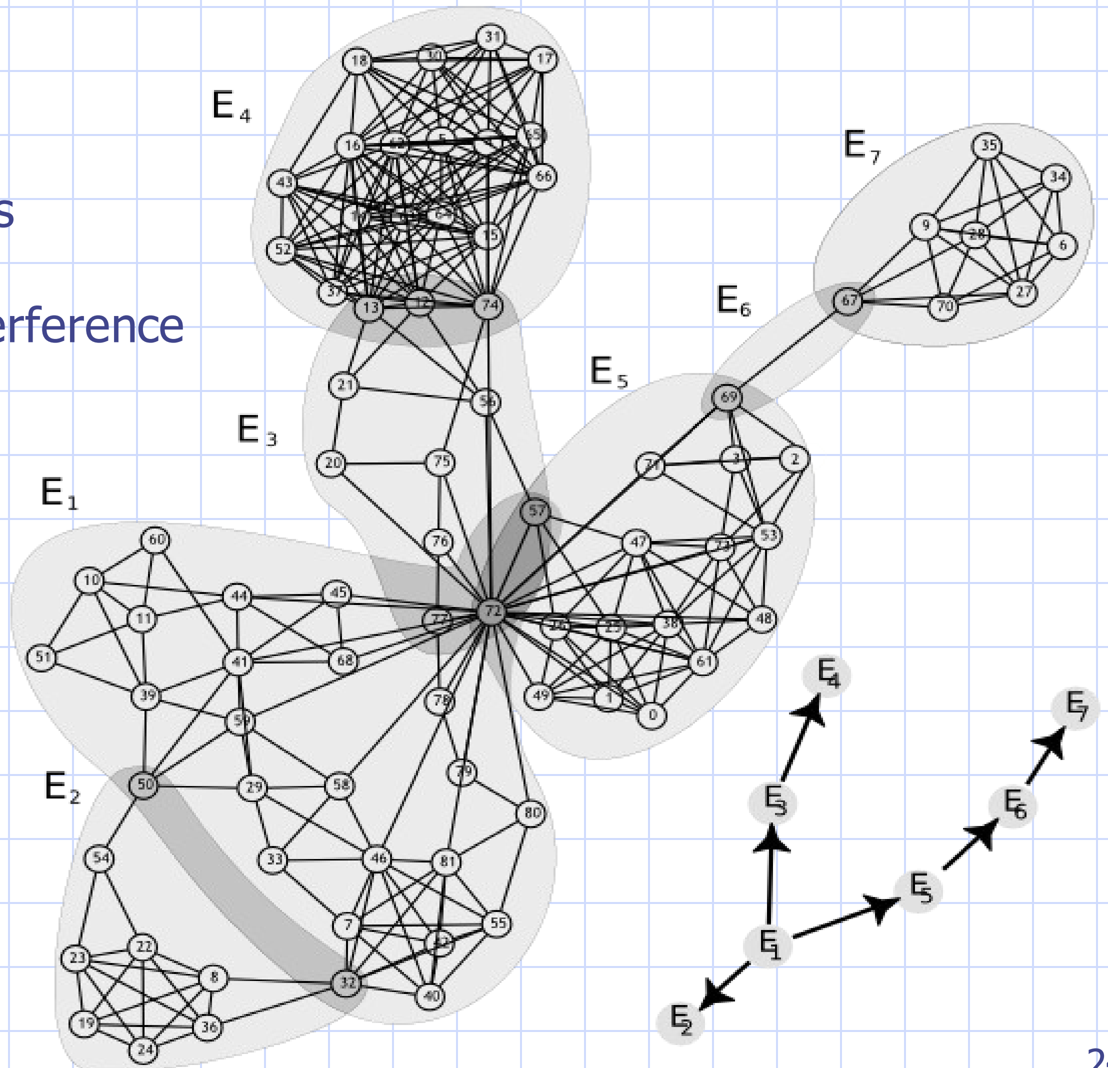
***CELAR** problem size: n=100—458 ; d=44 ; e=1,000—5,000*

Tree decomposition example

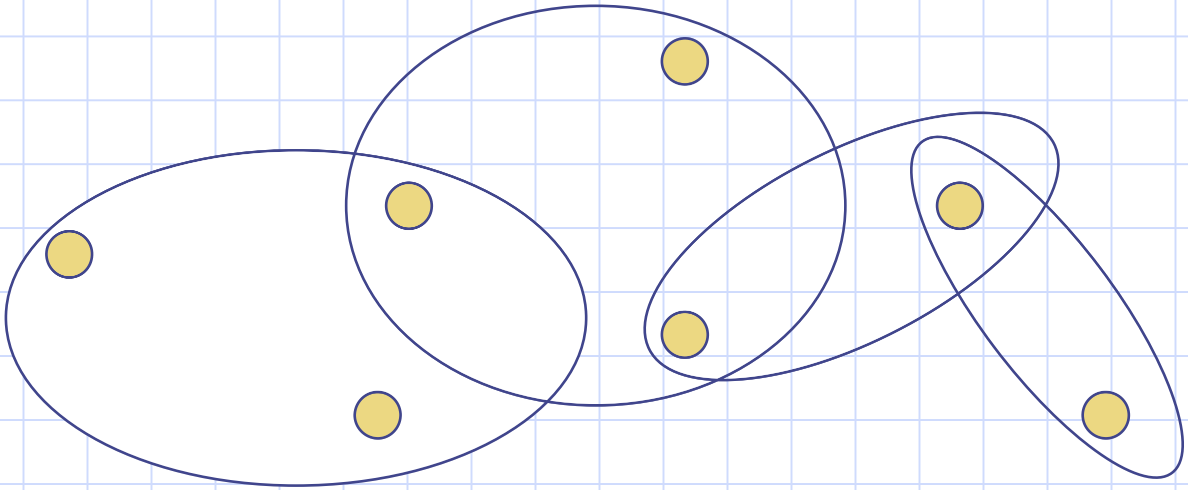
Benchmark problem
assigning frequencies
to transmitters
to minimise total interference

CELAR scen06r
n = 82
d = 44
e = 327

w = 26
s = 3

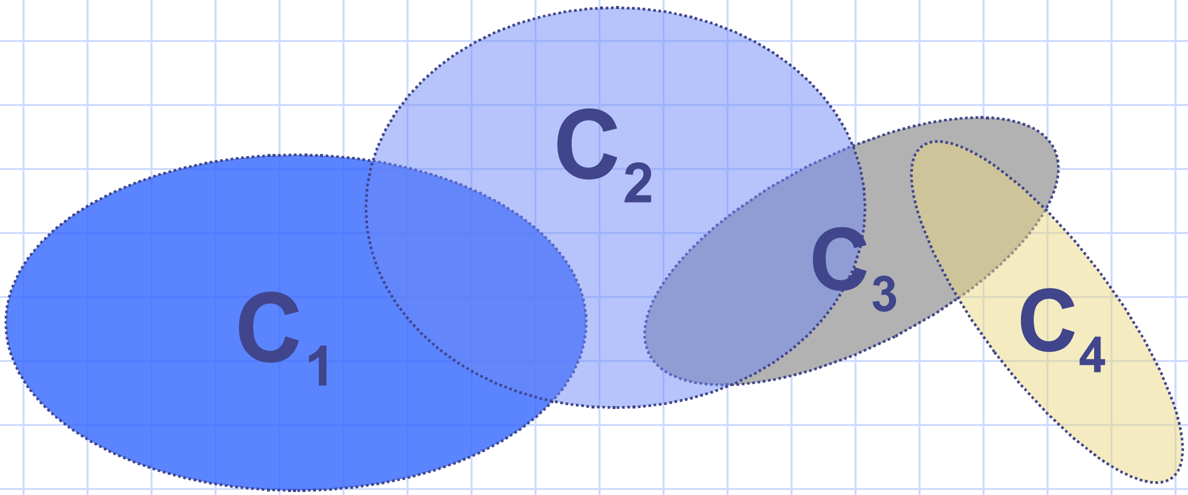


Problem features



- We have seen that structural features of a problem can lead to tractability
- This is very similar to the standard CSP
- What about other kinds of restrictions to the VCSP?

More problem features



- The picture now emphasises the cost functions
- Restricting the cost functions we allow can also lead to tractability

Valued constraint languages

- A set of cost functions is called a valued constraint language
- VCSP(Γ) represents the set of VCSP instances whose cost functions belong to the valued constraint language Γ
- For some choices of Γ, VCSP(Γ) is tractable
- We will consider some examples where the valuation structure contains non-negative real values and infinity, and aggregation is standard addition

Submodular functions

A class of functions that has been widely studied in OR is the submodular functions…

A cost function c is submodular if $\forall \mathbf{s},\mathbf{t}$

$$c(\min(\mathbf{s},\mathbf{t})) + c(\max(\mathbf{s},\mathbf{t})) \leq c(\mathbf{s}) + c(\mathbf{t})$$

where min and max are applied component-wise, i.e.

$$\min(\langle s_1,\ldots,s_k\rangle,\langle t_1,\ldots,t_k\rangle) = \langle \min(s_1,t_1),\ldots,\min(s_k,t_k)\rangle$$

VCSP($\Gamma_{\text{submodular}}$) is tractable

Examples of submodular functions

x	y	z	
0	0	0	0
0	0	1	1
0	1	0	7
0	1	1	1
1	0	0	∞
1	0	1	3
1	1	0	∞
1	1	1	0

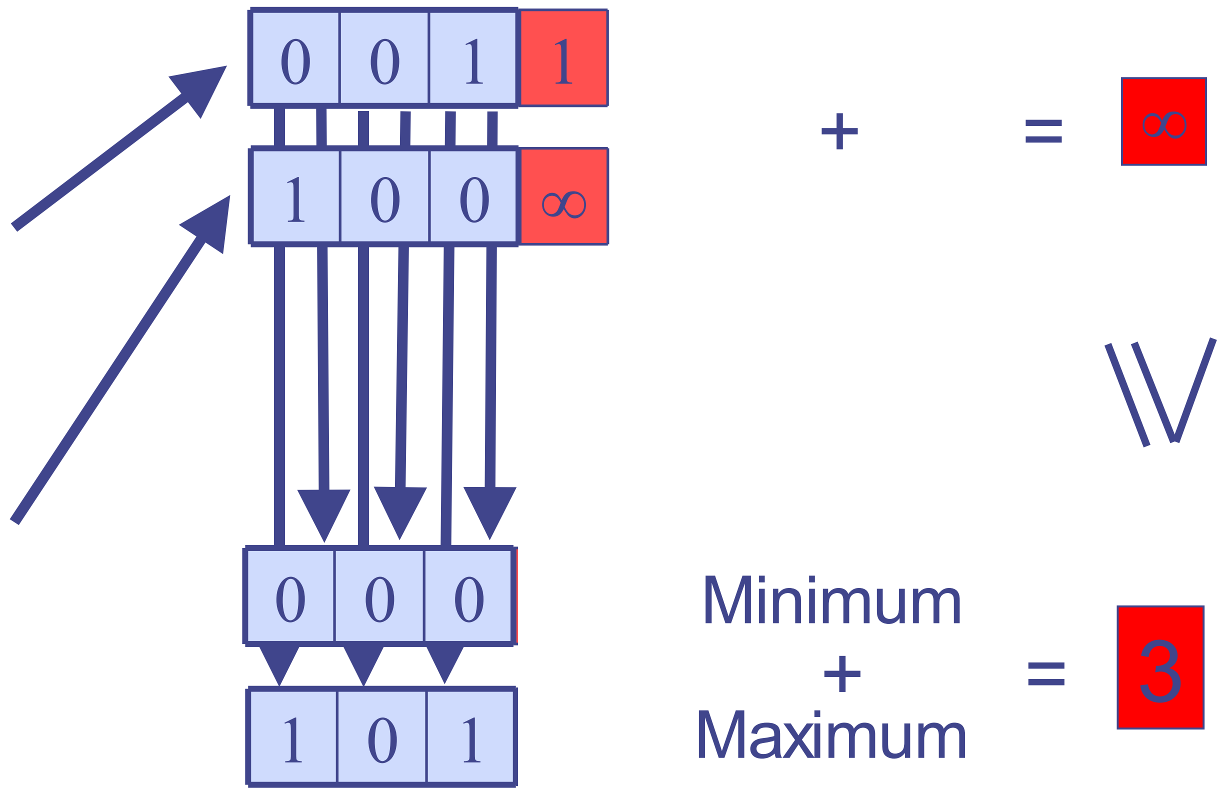
1	0	1

Maximum

Examples of submodular functions

$\forall s,t \;\; \mathrm{Cost}(\mathrm{Min}(s,t)) + \mathrm{Cost}(\mathrm{Max}(s,t)) \leq \mathrm{Cost}(s) + \mathrm{Cost}(t)$

	x	y	z	
	0	0	0	0
s	0	0	1	1
	0	1	0	7
	0	1	1	1
t	1	0	0	∞
	1	0	1	3
	1	1	0	∞
	1	1	1	0



Examples of submodular functions

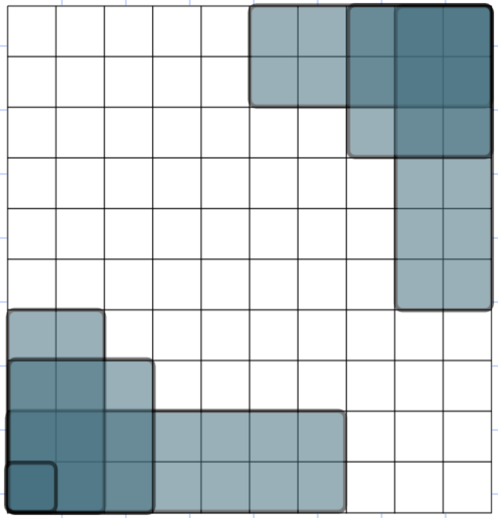
- all unary functions
- all linear functions (of any arity)
- the binary function ϕ_{cut}

  where $\phi_{cut}(a,b)=1$ if $(a,b)=(0,1)$ (0 otherwise)
- the rank function of a matroid
- the Euclidean distance function between two points $(x_1, x_2), (x_3, x_4)$ in the plane
- $\phi(x,y)=(x-y)^r$ if $x \geq y$ (∞ otherwise) for $r \geq 1$

(compare "*Simple Temporal CSPs with strictly monotone preferences*"
Khatib et al, IJCAI 2001)

Binary submodular functions

Binary submodular functions over any finite domain can be expressed as a sum of "Generalized Interval" functions



(they correspond to Monge matrices)

Binary VCSP($\Gamma_{\text{submodular}}$) is $O(n^3d^3)$

See Cohen et al "A maximal tractable class of soft constraints", JAIR 2004

Beyond submodularity

$$\forall s,t \;\; \mathrm{Cost}(\mathrm{Min}(s,t)) + \mathrm{Cost}(\mathrm{Max}(s,t)) \leq \mathrm{Cost}(s) + \mathrm{Cost}(t)$$

x	y	z	
0	0	0	0
0	0	1	1
0	1	0	7
0	1	1	1
1	0	0	∞
1	0	1	3
1	1	0	∞
1	1	1	0

The cost function has the ***multimorphism*** (Min,Max)

By choosing *other* functions, we can obtain other tractable valued constraint languages…

Known tractable cases

If the cost functions all have one of these eight multimorphisms, then the problem is tractable:

1) (Min,Max)
2) (Max,Max)
3) (Min,Min)
4) (Majority,Majority,Majority)
5) (Minority,Minority,Minority)
6) (Majority,Majority,Minority)
7) (Constant 0)
8) (Constant 1)

See Cohen et al “The complexity of soft constraint satisfaction”, AIJ 2006

A dichotomy theorem

If the cost functions all have one of these eight multimorphisms, then the problem is tractable:

1) (Min,Max)
2) (Max,Max)
3) (Min,Min)
4) (Majority,Majority,Majority)
5) (Minority,Minority,Minority)
6) (Majority,Majority,Minority)
7) (Constant 0)
8) (Constant 1)

For Boolean cost functions…

In all other cases the cost functions have **no** significant common multimorphisms and the VCSP problem is **NP-hard**.

See Cohen et al “The complexity of soft constraint satisfaction”, AIJ 2006

Benefits of a general approach

- The dichotomy theorem immediately implies earlier results for SAT, MAX-SAT, Weighted Min-Ones and Weighted Max-Ones

Bibliography

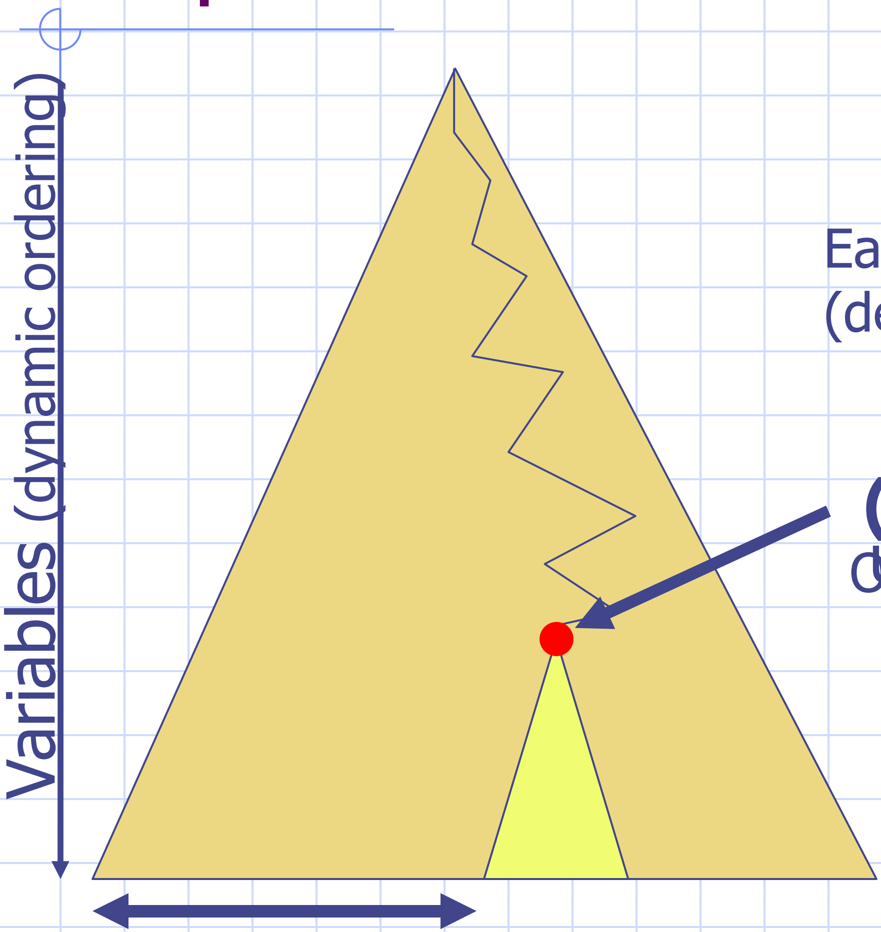
- For general background on tractable structures, see the chapter on "Tractable Structures" by Dechter, in the *Handbook of Constraint Programming*, Elsevier, 2006.

- For tractable valued constraint languages see "The complexity of soft constraint satisfaction", Cohen, Cooper, Jeavons & Krokhin, AIJ 2006.

Chapter 3. Search using problem transformations

Branch and Bound,
Equivalence-preserving operations,
Local consistency (node, arc, directional, virtual, optimal),
Global cost functions.

Depth-First Branch and Bound (DFBB)



Each node is a VCSP subproblem
(defined by current conditioning)

(LB) Lower Bound = $c_\varnothing$

under estimation of the best solution in the sub-tree

Obtained by enforcing local consistency

If LB $c_\varnothing$ ≥ *k* then prune

(UB) Upper Bound
= best solution found so far = k

Local consistency

A property that says that the network is "explicit" enough at a local level

Filtering algorithm: transforms a network in an equivalent network that satisfies the property (closure)

CSP: pol. time, confluent, incremental

Equivalence-preserving transformations (EPT)

- An EPT transforms VCSP instance P1 into another VCSP instance P2 with the same objective function.
- Examples of EPTs:

- Propagation of inconsistencies (∞ costs)
- UnaryProject
- Project/Extend

INCREMENTALITY!

UnaryProject(i,α)

Precondition: $0 \leq \alpha \leq \min\{c_i(a) : a \in d_i\}$

$c_0 := c_0 + \alpha$;

for all $a \in d_i$ **do**

$c_i(a) := c_i(a) - \alpha$;

Increases the lower bound c_0 if all unary costs $c_i(a)$ are non-zero.

Project(M,i,a,α)

Precondition: $i \in M$, $a \in d_i$, $-c_i(a) \leq \alpha \leq \min\{c_M(x): x[i]=a\}$

$c_i(a) := c_i(a) + \alpha$;

for all $x \in$ labelings(M) s.t. x[i]=a **do**

$c_M(x) := c_M(x) - \alpha$;

If $\alpha>0$, this projects costs from c_M to c_i

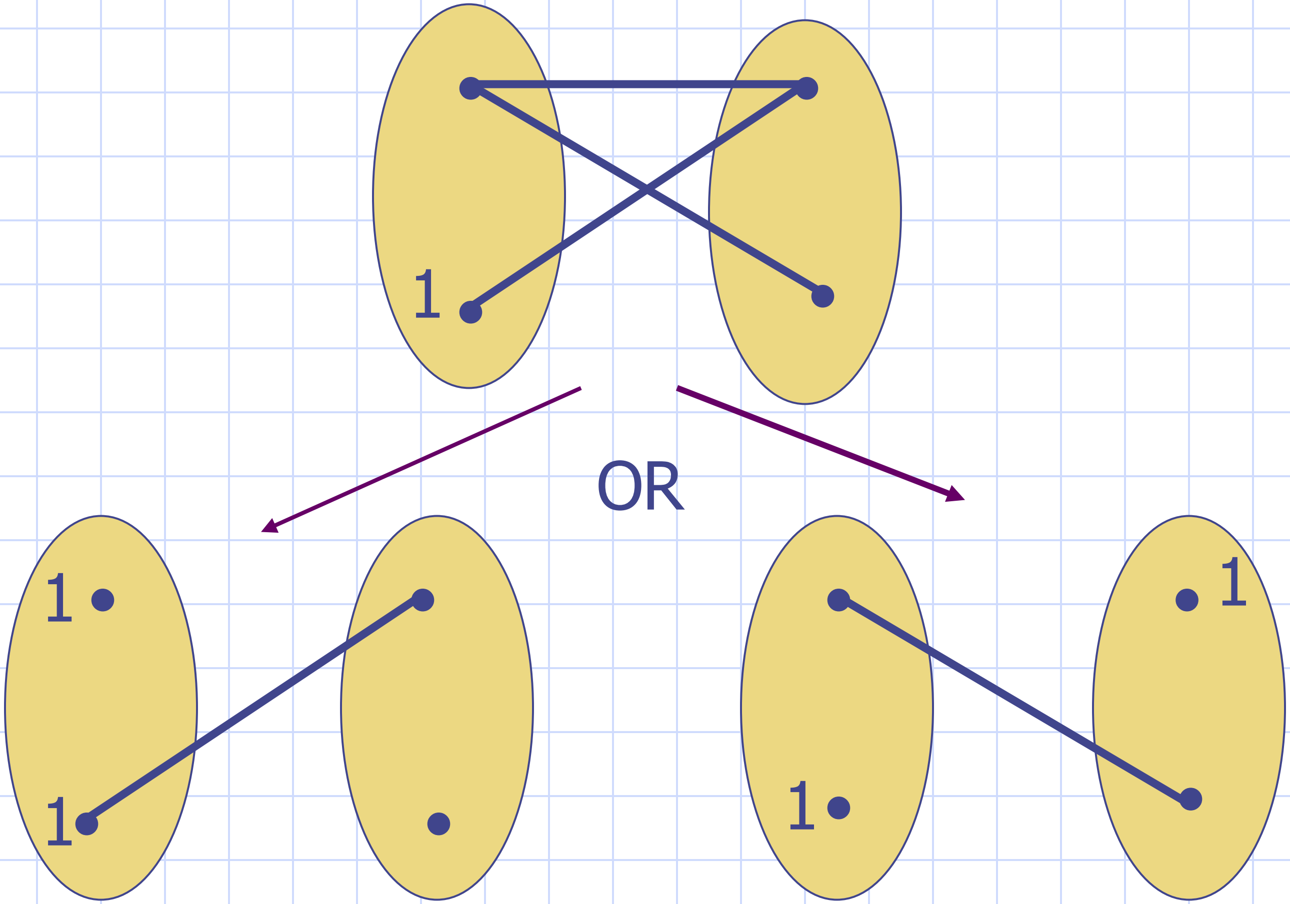
If $\alpha<0$, this extends costs from c_i to c_M

Node and soft arc consistency

- Node consistent (NC) if $\forall i$
  no UnaryProject(i,α) is possible for $\alpha > 0$ and no propagation of ∞ costs possible between c_i and c_0 (forbidden values removed if $c_i + c_0 \geq k$)

- Soft arc consistent (SAC) if $\forall M,i,a$
  no Project(M,i,a,α) is possible for $\alpha > 0$

The SAC closure is not unique



Finding the best order of integer EPT application is NP-hard (Cooper, Schiex 2004)

Different soft AC notions:

- Directional: send costs from X_j to X_i if $i<j$ (in the hope that this will increase c_0)
- Existential: $\forall i$, send costs to X_i simultaneously from its neighbor variables if this increases c_0
- Virtual: no *sequence* of Projects/Extends increases c_0
- Optimal: no *simultaneous set* of Projects/Extends increases c_0

Directional Arc Consistency

- for all $i<j$, $\forall a \in d_i \; \exists b \in d_j$ such that $c_{ij}(a,b) = c_j(b) = 0$.
- Solves tree-structured VCSPs
- FDAC (Full Directional AC) =
  Directional AC + Soft AC
- FDAC can be established in $O(end^3)$ time (or in $O(ed^2)$ time if $+_k$ is $+$)

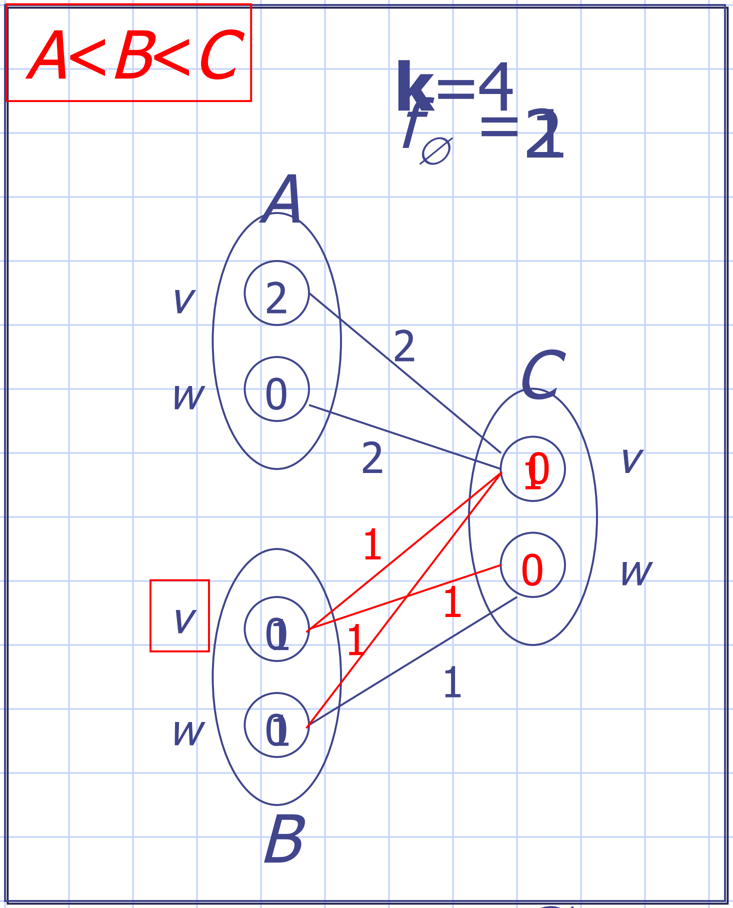
Directional AC (DAC*)

- NC*
- For all f_{AB} $(A<B)$
  - $\forall a \exists b$

    $f_{AB}(a,b) + f_B(b) = 0$

- b is a *full-support*
- complexity:

  $\mathbf{O}(ed^2)$



Shift(f_{BC},(C,v),-1) Shift(f_{BC},(B,v),1) Shift(f_{BC},(B,w),1) Shift(f_B, $\varnothing$,1)

Shift(f_A,$\varnothing$, -2) Shift(f_A,$\varnothing$, 2)

Existential Arc Consistency

- node consistent and $\forall i, \exists a \in d_i$ such that $c_i(a) = 0$ and for all cost functions c_{ij}, $\exists b \in d_j$ such that $c_{ij}(a, b) = c_j(b) = 0$
- EDAC = Existential AC + FDAC
- EDAC can be established in
  $O(ed^2 \max\{nd,k\})$ time

Virtual Arc Consistency (VAC)

(Cooper et al, 2008)

- If P is a VCSP instance then Bool(P) is the CSP instance whose allowed tuples are the zero-cost tuples in P-c_0
- If Bool(P) has a solution, then P has a solution of cost c_0 (but usually Bool(P) has no solution)
- Definition: P is VAC if Bool(P) is AC.

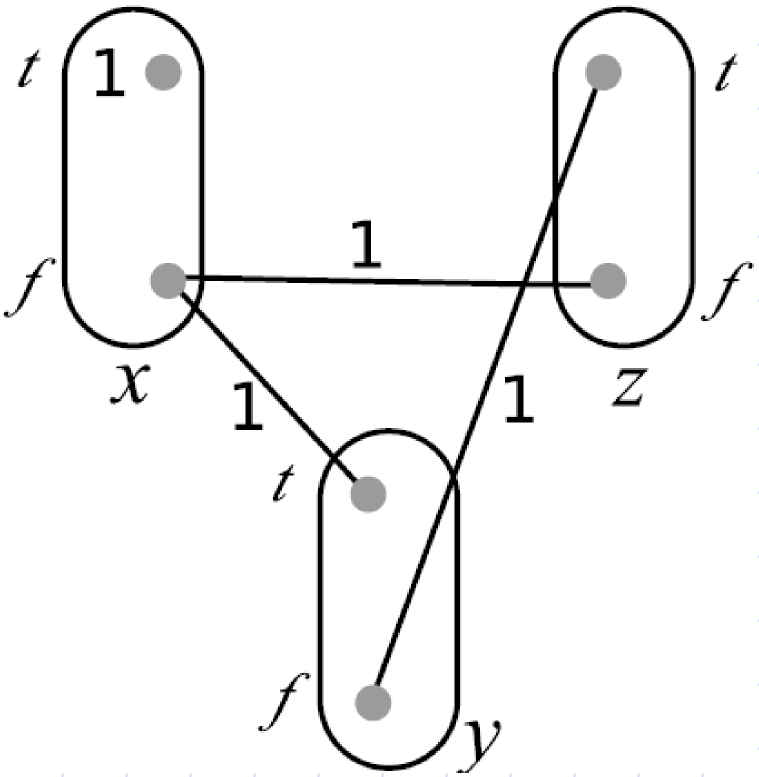
Approximating VAC

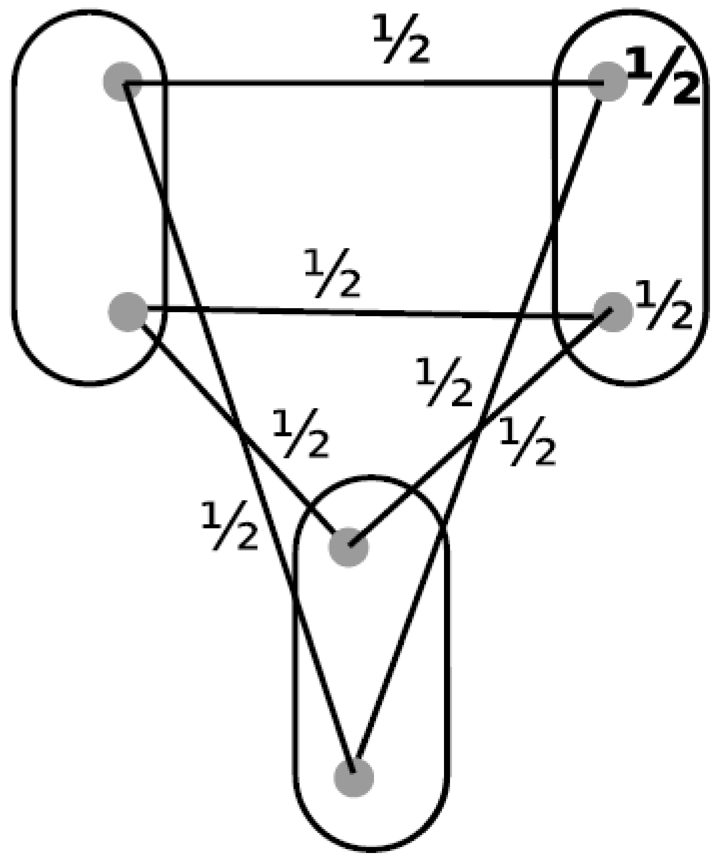
(similar to Augmenting DAG, Schlesinger et al, 30 years before)

- If a sequence of AC operations in Bool(P) leads to a domain wipe-out, then a similar sequence of SAC operations in P increases c_0
- But, in this sequence, costs may need to be sent in more than one direction from the same c_M $\Rightarrow$ Introduction of *fractional* weights
- VACε algorithm may converge to a local minimum (and more, an instance P′ which is *not* VAC)
- VACε can be established in $O(ed^2\ k/\varepsilon)$ time

Enforcing VAC

AC,DAC,FDAC,EDAC





Optimal Soft Arc Consistency

(Cooper et al. 2007), similar to (Schlesinger, 30 years before)

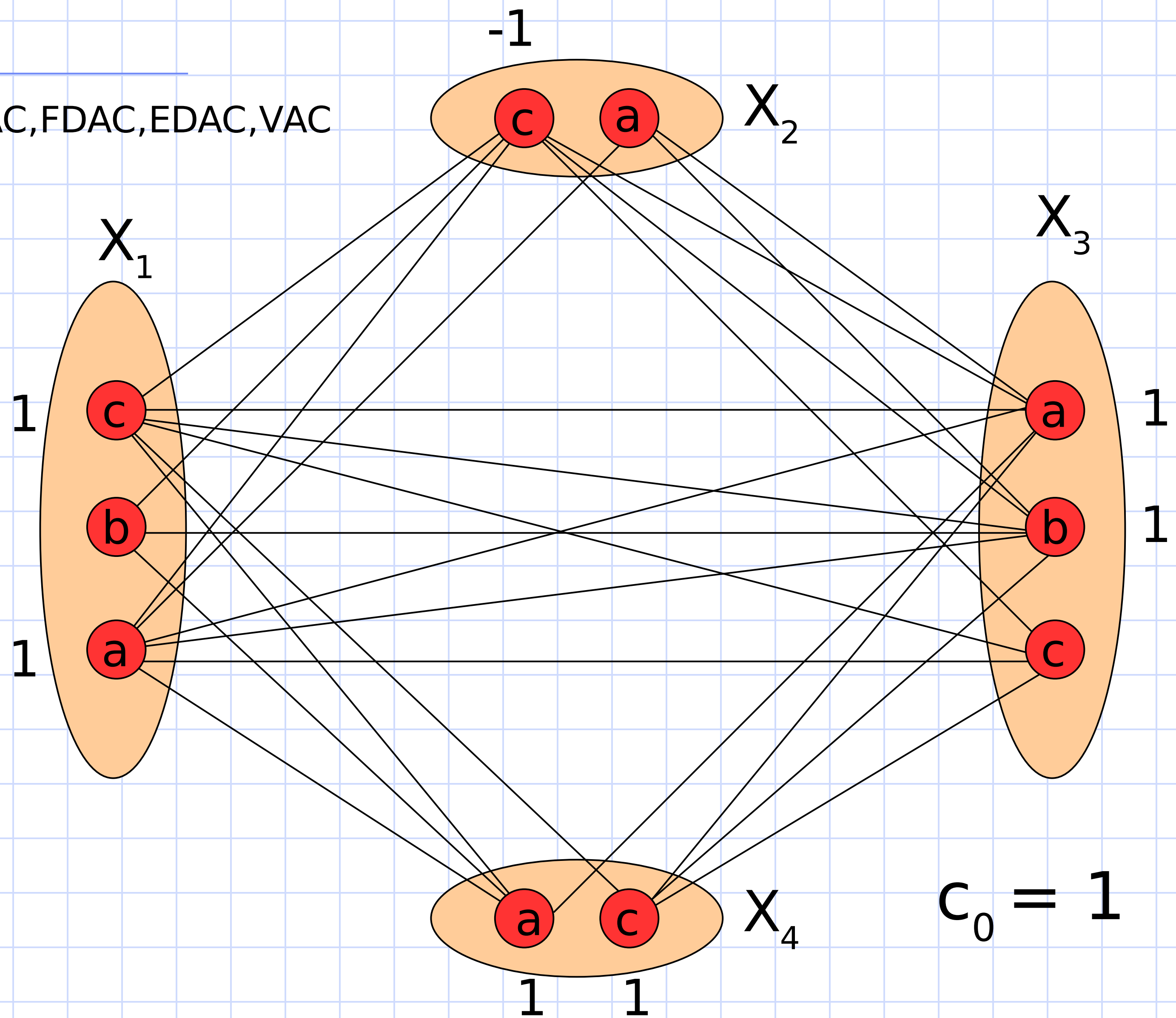
- We can overcome this problem of convergence by solving a LP to find the *set* of simultaneous UnaryProject and Project operations which maximises c_0.

- The resulting VCSP instance is OSAC (Optimal Soft Arc Consistent).
- OSAC is strictly stronger than VAC.
- Unfortunately, the LP has O(edr+n) variables and O(ed^r+nd) constraints(pre-processing).

$$p^{23}_{2c} = p^{34}_{3a} = p^{31}_{3b} = p^{12}_{1a} = p^{14}_{1c} = -1$$

$$p^{23}_{3a} = p^{23}_{3b} = p^{34}_{4c} = p^{31}_{1a} = p^{31}_{1c} = p^{12}_{2c} = p^{14}_{4a} = u_4 = 1$$

AC,DAC,FDAC,EDAC,VAC



Virtual Arc Consistency solves (locally-defined) submodular VCSP

If P $\in$ VCSP(Γ_{sub}) and P is VAC,
then Bool(P) is arc consistent, max-closed.

Hence, Bool(P) has a solution.

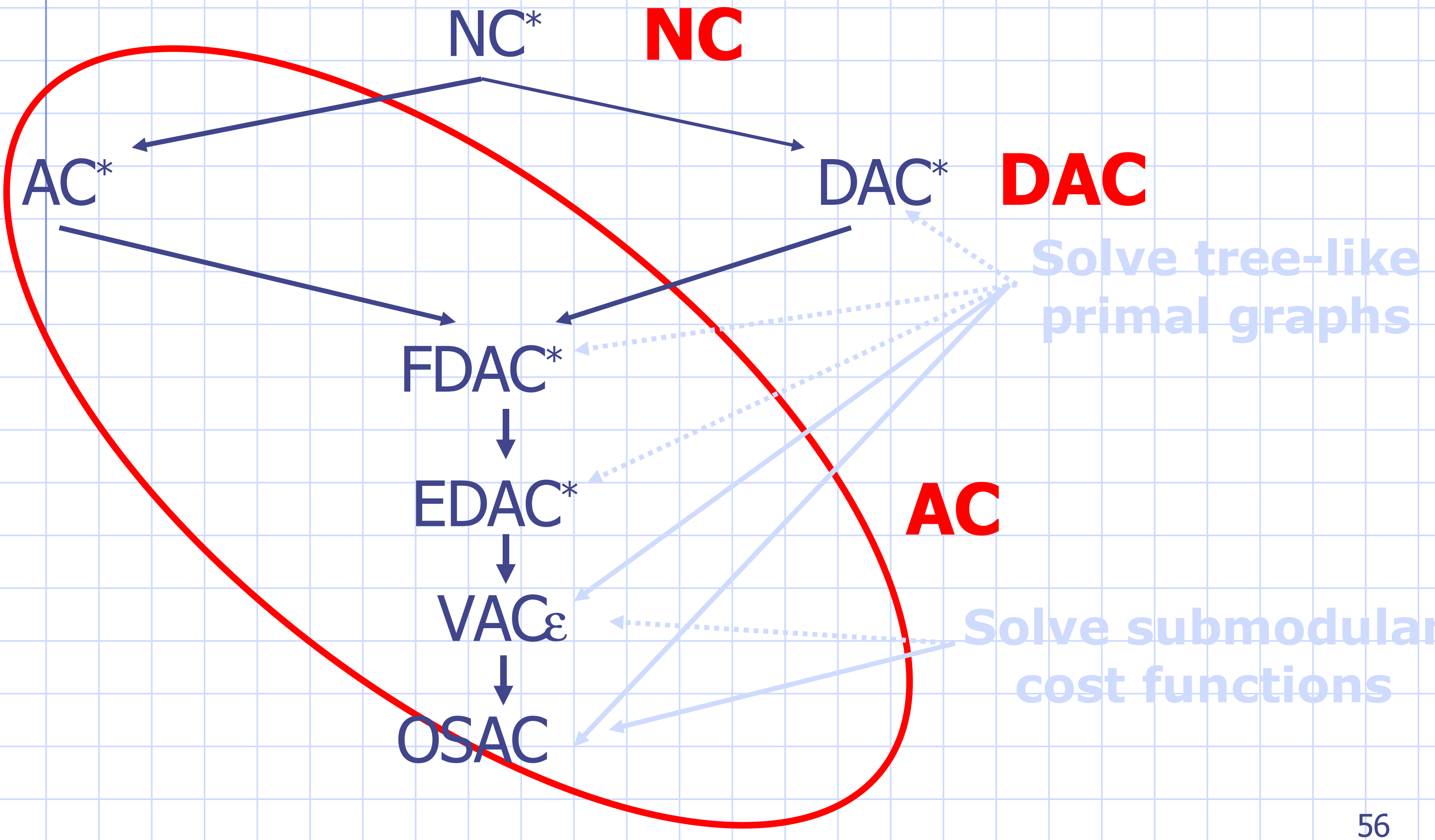
This solution has cost c_0 in P and is thus necessarily optimal.

Thus OSAC solves SFM since Project and UnaryProject preserve submodularity.

Also permutated submodular (some technicalities)

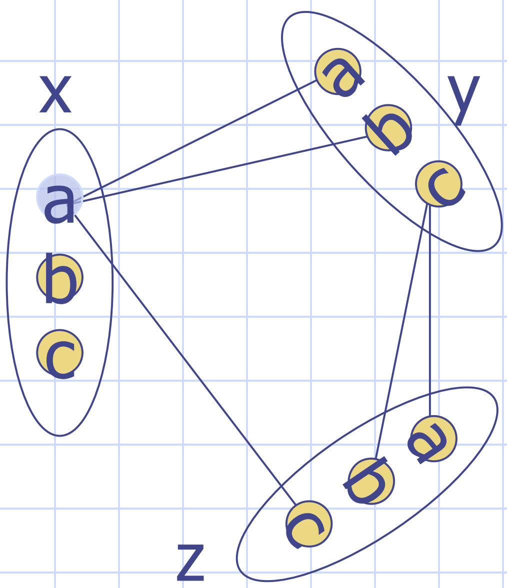
Hierarchy

Special case: CSP (**k**=1)



Beyond Arc Consistency

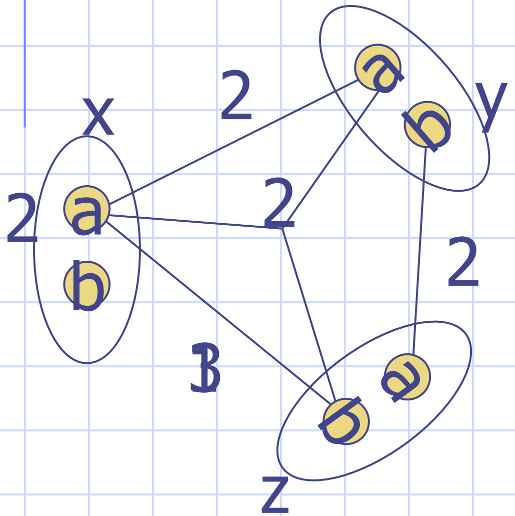
- Path inverse consistency (Debryune & Bessière)



(x,a) can be pruned because there are two other variables y,z such that (x,a) cannot be extended to any of their values.

Beyond Arc Consistency

- Soft Path inverse consistency

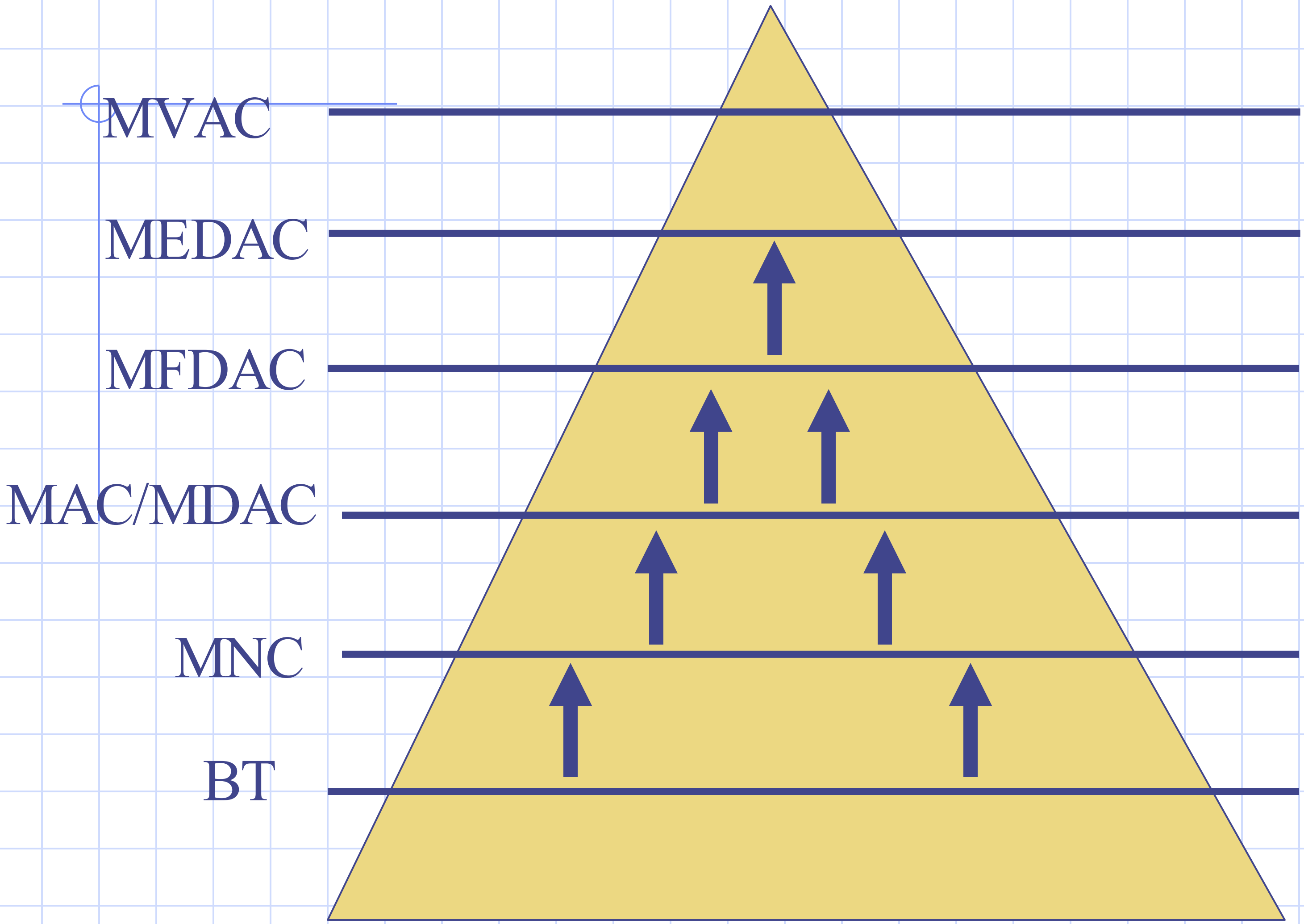


$f_y \oplus f_z \oplus f_{xy} \oplus f_{xz} \oplus f_{yz}$

x	y	z	
a	a	a	0
a	a	b	3
a	b	a	0
a	b	b	1
b	a	a	0
b	a	b	0
b	b	a	2
b	b	b	0

$(f_y \oplus f_z \oplus f_{xy} \oplus f_{xz} \oplus f_{yz})[x]$

x	
a	2
b	0


MVAC
MEDAC
MFDAC
MAC/MDAC
MNC
BT

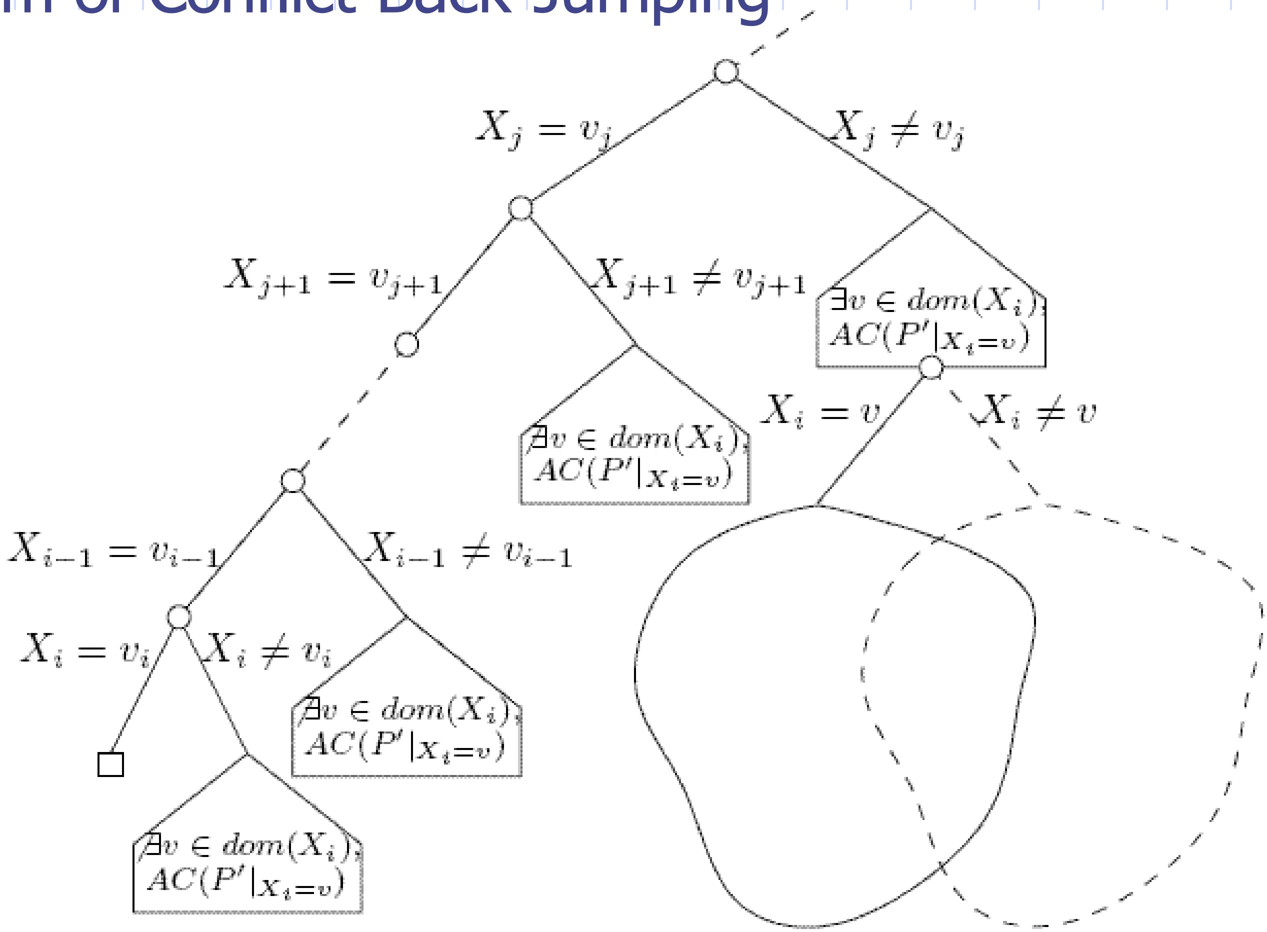
Some practical observations

- For very hard-to-solve instances, maintaining VAC provides a significant speed-up (closed RLFAP graph11/13).
- For many problems, maintaining a simpler form of soft arc consistency (e.g. EDAC) is faster.
- Unary costs $c_i(a)$ and EAC value inform value and variable ordering heuristics

Variable heuristic

Last-conflict (Lecoutre et al, ECAI 2006)

Basic form of Conflict Back-Jumping



Used in combination with domain size / weighted degree (Lecoutre et al, AIJ 2009), breaking ties with max unary cost

RLFAP: CELAR 6 results since 1993

n. of vars: n=100, domain size: d=44, n. of cost functions: e=1222

Time of optimality proof		Method(s) used	Publication
26 days **(SUN UltraSparc 167 MHz)**		Ad-hoc problem decomposition & Russian Doll Search *(22 vars only)*	(de Givry, Verfaillie, Schiex, CP 1997)
3 days (SUN Sparc 2)		Ad-hoc problem decomposition & PFC-MRDAC *(22 vars only)*	(Larrosa, Meseguer, Schiex, AIJ 1998)
8 hours (DEC Alpha 500MP)		Preprocessing rules & BbB Elimination	(Koster PhD thesis, 1999)
3 hours (PC 2.4 GHz)		B&B with EDAC & tree decomposition (BTD)	(de Givry, Schiex, Verfaillie, AAAI 2006)
1' 26'' **(PC 2.5 GHz)**	**25000x** **16 x**	BTD-RDS & variable ordering heuristics & dichotomic branching	(Sanchez, Allouche, de Givry,Schiex, IJCAI 2009)

CELAR 7 (n=200) solved in 4.5 days (Sanchez et al, IJCAI 2009)

CELAR 8 (n=458) solved in < 2 days (127 days)

2010 Approximate Inference Evaluation (results given at UAI'10)

Networks – by domain (1 hour)

Network	PR	MAR	MPE
CSP	8	8	55
Grids	20	20	40
Image Alignment			10
Medical Diagnosis	26	26	
Object Detection	96	96	92
Pedigree	4	4	
Protein Folding			21
Protein-Protein Interaction			8
Segmentation	50	50	50

Summary of the results

Seconds	PR	MAR	MPE
20	Arthur Choi (UCLA)	Arthur Choi (UCLA)	Joris Mooij (Max Planck)
1200	Vibhav Gogate (UW+UCI)	Vibhav Gogate (UW+UCI)	Thomas Schiex (INRA)
3600	Vibhav Gogate (UW+UCI)	Vibhav Gogate (UW+UCI)	Joris Mooij (Max Planck)

toulbar2 was also first at UAI'08 Evaluation, MaxCSP'06,'08 Competition

Winning Teams

- (MAR) **IJGP** by Vibahv Gogate (UW), Andrew Gefland, Natasha Flerova and Rina Dechter (UCI):
  Anytime iterative GBP based algorithm
- (PR) **Vgogate** by Vibahv Gogate, Pedro Domingos (UW),, Andrew Gefland and Rina Dechter (UCI):
  Formula based importance sampling
- (PR+MAR) **EDBP** by Arthur Choi, Adnan Darwiche, with support from Glen Lenker and Knot Pipatsrisawat (UCLA):
  Anytime BP based anytime thickening of structure
- (MAP) **libDAI** by Joris Mooij (Max Planck):
  junction tree, LBP/MP, double-loop GBP, Gibbs, decimation
- (MAP) **toulbar2** by Thomas Schiex et al (INRA)
  Anytime branch and bound weighted CSP solver

73% instances solved exactly

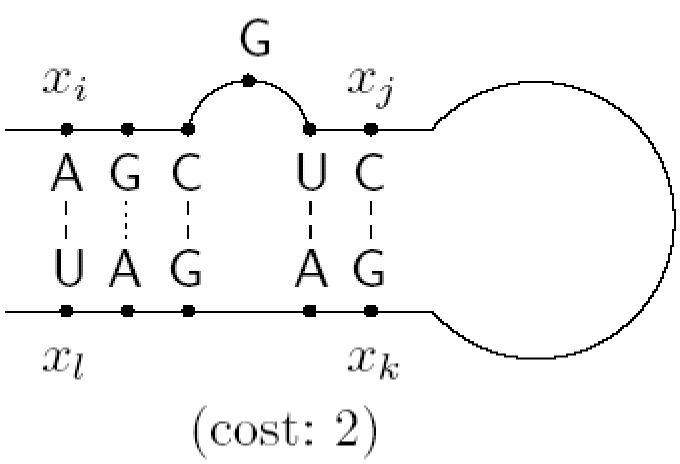
AC for global cost functions

- **Global constraints**: specific family of constraints on an unbounded number of variables with efficient local consistency filtering.
  - Example: AllDifferent (max matching)

- Same for global cost functions
  - Example: # of variables with the same value

    (van Hoeve et al, J. Heur. 2006) (Lee & Leung, IJCAI'09)

RNA gene finding (Zytnicki et al, 2008)



- Given a sequence and an RNA gene descriptor
- …find all the occurrences of the descriptor with at most *k* mismatches
  - NP-complete for *k*=0 (Vialette, 2004)
  - Sort solutions by their number of mismatches

***RNA** problem sizes: n=20 ; d>100 million! ; e(4)=10*

Bound arc consistency

- Goal: space complexity independent of the domains
- BAC$^{\varnothing}$ (Zytnicki et al, *JAIR, 2010*)
  - Avoid EPTs, except those shifting cost to
  - Prune *extremity* domain values only
  - Complexity
    - Time $\mathbf{O}(n^2 r^2 d^{r+1})$ and space $\mathbf{O}(n+er)$

    with maximum constraint arity r
  - BAC$^{\varnothing}$ is confluent
  - Can be specialized for semi convex cost functions (d to d^2 speedup on binary CF)

RNA gene finding

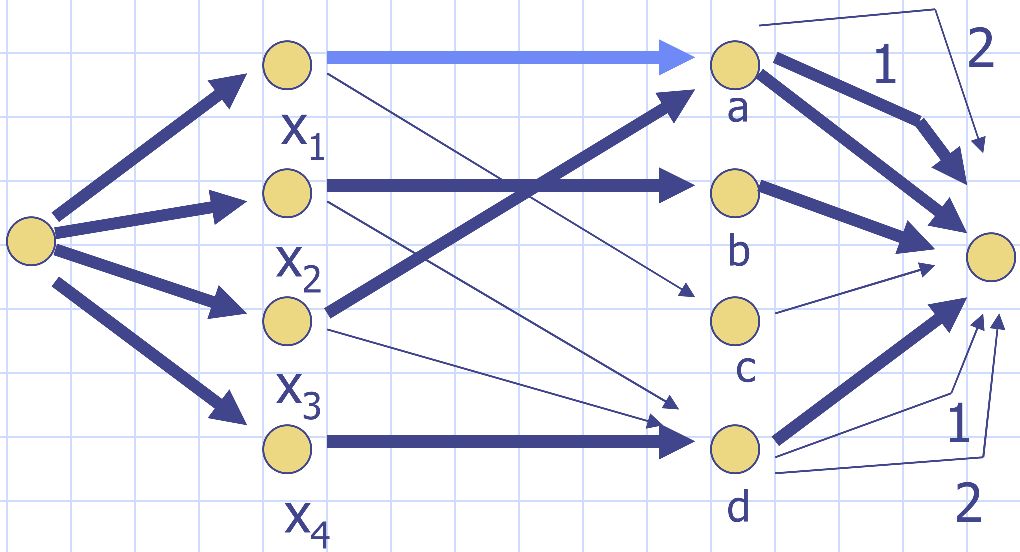
Domain size	10k	50k	100k	500k	1M	4.9M
Nb. of solutions	32	33	33	33	41	274
AC*						
Time	1hour 25min.	44hours	-	-	-	-
Nb. of backtracks	93	101	-	-	-	-
BAC						
Time (sec.)	0.016	0.036	0.064	0.25	0.50	2.58
Nb. of backtracks	93	101	102	137	223	1159

Fig. 6. Searching all the solutions of a tRNA motif in *Escherichia coli* genome.

Darn! solver (Zytnicki et al, Constraints 2008)

Network representing "min number of variables with same value"

(Beldiceanu & Petit, CPAIOR'04)



All edge capacities are equal to 1

All edge costs are 0 if not indicated

Flow shown is a min-cost max-flow with x_1=a.

We can project 1 from c_M to $c_1(a)$ by reducing the cost of the light blue edge from 0 to –1.

Latin Square N x N with costs

Example of solution for N = 5:

2	1	3	5	4
4	2	1	3	5
1	5	4	2	3
5	3	2	4	1
3	4	5	1	2

All variables take a different value in each row and each column

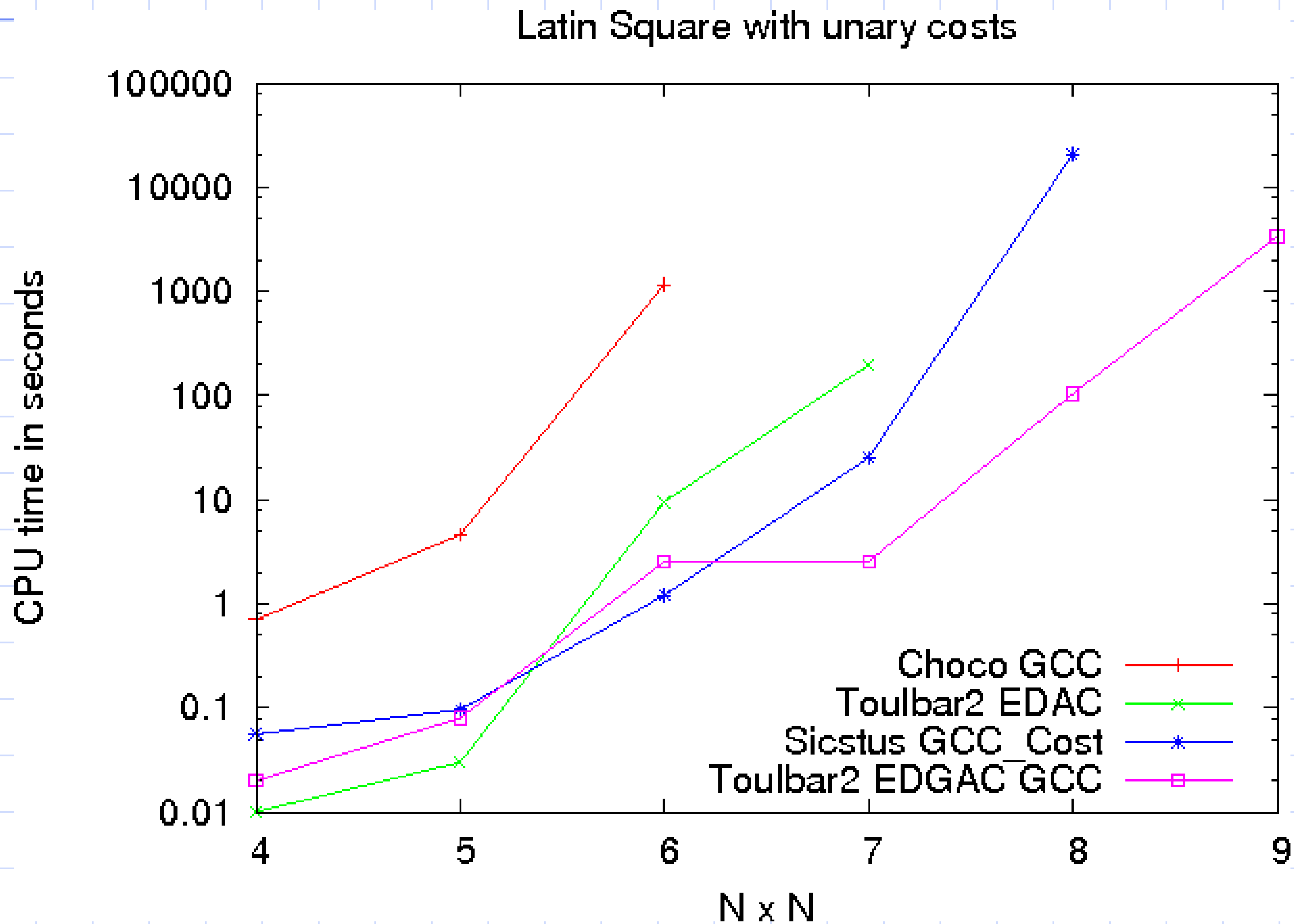
A unary cost function for each cell
$f^{i,j}(x_{i,j}) : D \rightarrow [0,MaxCost[$

Objective: 49

$$\text{Objective} = \sum_i \sum_j f^{i,j}(x_{i,j})$$

GCC_Cost (Régin, *Constraints* 2002)
EDGAC (Lee & Leung, *AAAI* 2010)

Latin Square with costs



choco v2.1.1, toulbar2 v0.9.3, sicstus v4.1.2 on linux PC 2.66 Ghz 64GB

Bibliography

- For an overview of soft local consistencies, see "*Soft arc consistency revisited*", Cooper, de Givry, Sanchez, Schiex, Zytnicki & Werner, AIJ 2010.

- For soft global constraints (FDGAC), see "*Towards Efficient Consistency Enforcement for Global Constraints in Weighted Constraint Satisfaction*", Lee & Leung, IJCAI 2009.

Chapter 4. Open problems

Concerning problem definition, search, transformations, tractable classes

Possible extensions to VCSP

- Partial order instead of total order
- 2 arbitrary binary operators (e.g. calculating the sum of products instead of the min of the sum subsumes #CSP)
- Objective function not constructible using a binary aggregation operator (e.g. the median of the set of costs)

Tractability

- Can we characterize/unify all tractable classes of VCSP over non-Boolean domains?
- Are there interesting tractable classes apart from submodular functions?
- Are there more efficient algorithms for submodular function minimisation?

New problem transformations

- Global cost functions
- Decomposition into several problems whose sum is equal to the original VCSP
- Transformations which preserve at least one solution (if it exists) but do not necessarily all costs (substituability).
- Applying rules involving ≥2 constraints

Conclusion

- VCSP combines CSP and optimisation in a unified way
- Technology is usable and useful, and still maturing
- Something different: Structure estimation in Gene Networks